Semantic Interoperability in Healthcare Information for EHR Databases

Shelly Sachdeva and Subhash Bhalla

Graduate Department of Computer and Information Systems
University of Aizu, Fukushima, Japan
{d8111107,bhalla}@u-aizu.ac.jp

Abstract. Healthcare information is complex, distributed and non-structured in nature. Integration of information is important to retrieve patient history, for knowledge sharing and to formulate queries. Large scale adoption of electronic healthcare applications requires semantic interoperability. Interoperability of Electronic Health Records (EHRs) is important because patients have become mobile, treatment and health care providers have increased, and also, have become more specialized. The paper analyses the role of semantic interoperability in healthcare. The system modeling approach has been analyzed with a view of supporting system-to-system and user-system interactions. In addition, query interfaces have been considered at varying levels of user and system activities.

Keywords: Electronic Health Records, Semantic Interoperability, openEHR, Healthcare, Archetype Based EHR.

1 Introduction

Digitized form of individual patient's medical record is referred as electronic health record (EHR). These records can be stored, retrieved and shared over a network through enhancement in information technology. An Integrated Care EHR is defined as:" a repository of information regarding the health of a subject of care in computer processable form, stored and transmitted securely, and accessible by multiple authorized users" [1]. The Institute of Electrical and Electronics Engineers (IEEE) defines interoperability as the "ability of two or more components to exchange information and to use the information that has been exchanged" [2]. Semantic Interoperability is a big challenge in healthcare industry. Semantic interoperability states that the meaning of information must be preserved from a user level through logical level to physical level. Users enter information. It should safely reach the designated part of the system, and allow it to be sharable with other users and systems. As we know, there are different vendors for different systems. Thus, Semantic interoperability should be taken into account during exchange of data, information and knowledge. Figure1 indicates that, the meaning of information will be preserved across various applications, systems and enterprises.

Health care domain is complex. It is evolving at a fast rate. Health knowledge is becoming broad, deep and rich with time. Often, different clinics and hospitals have

S. Kikuchi, S. Sachdeva, and S. Bhalla (Eds.): DNIS 2010, LNCS 5999, pp. 157–173, 2010.

their own information systems to maintain patient data. There may be redundancy in data because of distributed and heterogeneous data resources. This may hinder the exchange of data among systems and organizations. There is a need for legacy migration of data to a standard form for the purposes of exchanges. The EHRs should be standardized and should incorporate semantic interoperability. World Health Organization (WHO) has strong desire to develop and implement semantically interoperable health information systems and EHRs [25]. The rest of paper is organized as follows. Section 2 describes the role of dual level modeling approach in achieving semantic interoperability. Section 3 describes the openEHR architecture and its comparison to database management system architecture. Section 4 explains the details of semantic interoperability in EHRs systems. Section 5 describes querying EHR data with emphasis on high-level query interfaces for health professionals. Section 6 describes the discussions. Finally, section 7 presents the summary and conclusions.

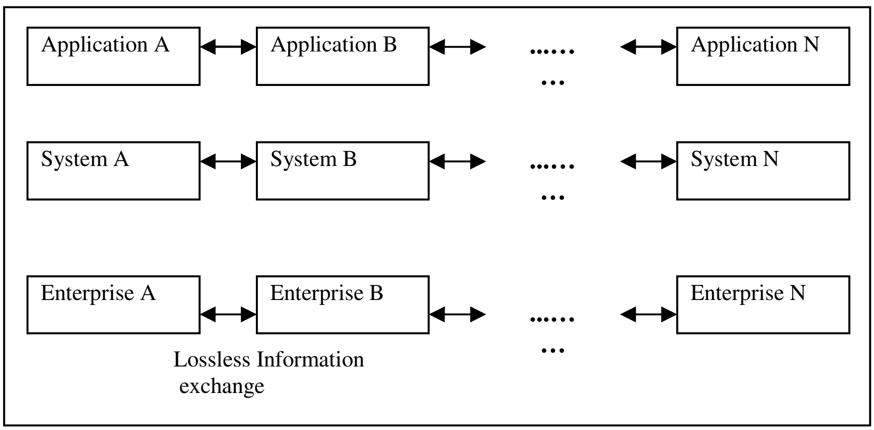

Fig. 1. Semantic Interoperability

2 Dual Level Modelling Approach

In essence, the proposed Electronic Health Records (EHRs) have a complex structure that may include data of about 100-200 parameters, such as temperature, blood-pressure and body mass index. Individual parameters have their own contents (Figure 2). In order to serve as an information interchange platform, EHRs use archetypes to accommodate various forms of contents [6]. The EHR data has multitude of representations. The contents can be structured, semi-structured or unstructured, or a mixture of all three. These can be plain text, coded text, paragraphs, measured quantities with values and units, date, time, date-time, and partial date/time, encapsulated data (multimedia, parsable content), basic types (such as boolean, state variable), container types (list, set) and uniform resource identifiers (URI).

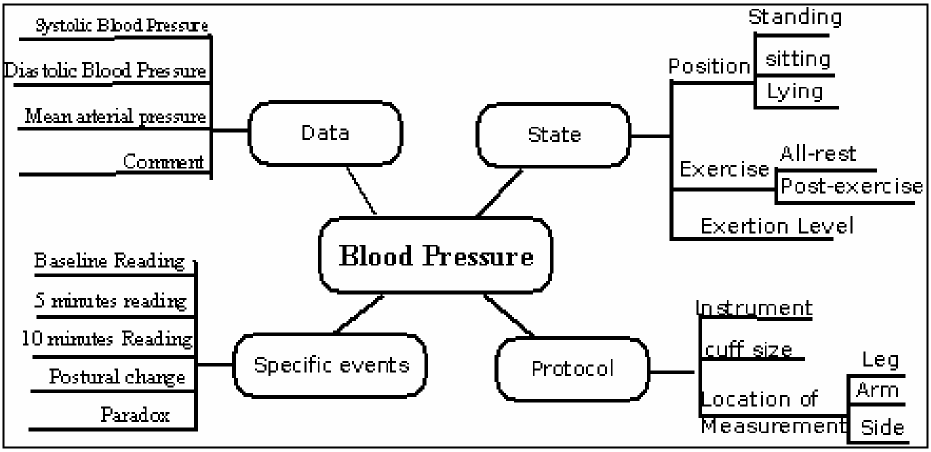

Fig. 2. Blood Pressure Archetype

The classic specifications require that the details of clinical knowledge be simultaneously coded into the software. The disadvantage of the approach has been that with the expansion of clinical knowledge the software becomes unsuitable and outdated. To overcome this, two level modeling approach has been proposed. Dual model approach for EHR architecture is defined by ISO 13606[12] for a single subject of care (patient). The emphasis is on achieving interoperability of systems and components during communication of a part or complete EHR. Examples of Dual Model EHR architecture are CEN/TC251 EN13606 [3] standard (developed by European committee for standardization) and openEHR standard. CEN/TC251 is a regional Standards Development Organization, which is addressing the needs of the stakeholders to have interoperable and implementable standards. It will allow for safe and secure information exchange. OpenEHR [4] foundation was established by University College London and Ocean informatics. It is an international foundation working towards semantic interoperability of EHR and improvement of health care.

In two-level modeling approach, the lower level consists of reference model and the upper level consists of domain level definitions in the form of archetypes and templates. Reference Model (RM) [12] is an object-oriented model that contains the basic entities for representing any entry in an EHR. The software and data can be built from RM. Concepts in openEHR RM are invariant. It comprises a small set of classes that define the generic building blocks to construct EHRs. This information model ensures syntactic or data interoperability. The second level is based on archetypes [5] [10]. These are formal definitions of clinical concepts in the form of structured and constrained combinations of the entities of a RM. A definition of data as archetypes can be developed in terms of constraints on structure, types, values, and behaviors of RM classes for each concept, and in the domain in which we want to use. Archetypes are flexible. They are general-purpose, reusable and composable. These provide knowledge level interoperability, i.e., the ability of systems to reliably communicate with each other at the level of knowledge concepts. Thus, the meaning of clinical data will be preserved in archetype based systems. Standardization can be achieved, so that, whenever there is a change in the clinical knowledge (or requirements), the software need not be changed and only the archetypes need to be modified.

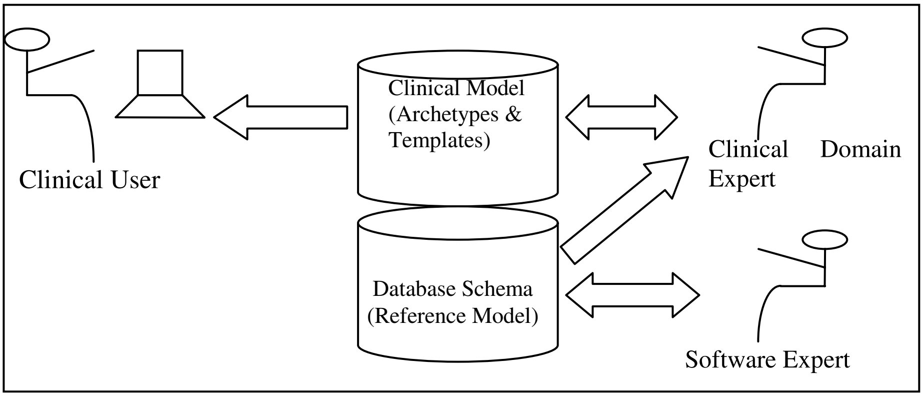

Fig. 3. Two Level Modelling Approach

The clinical user can enter and access information through clinical application. The clinical domain expert can record and maintain the clinical model through modeller. The modeller is software needed to manage the archetypes. Patients can have complete control over access and distribution of their health records.

3 The OpenEHR Architecture

The openEHR is pioneering the field for maintaining semantic interoperable EHRs. It has launched the implementation of the specification project. It aims at a new business model for electronic medical records. The latest edition of Microsoft's Connected Health Framework, includes openEHR (ISO 13606-2) archetypes as part of its domain knowledge architecture. The openEHR Reference Model is based on ISO and CEN EHR standards, and is interoperable with HL7 (Health Level Seven) and EDIFACT (Electronic data interchange for administration, commerce and transport) message standards [12]. This enables openEHR-based software to be integrated with other software and systems. Figure 4 shows how the DBMS architecture [7] can be compared to the openEHR architecture [6].

i) Physical level: The lowest level of abstraction describes the details of reference model. These include identification, access to knowledge resources, data types and structures, versioning semantics, support for archetyping and semantics of enterprise level health information types.

ii) Logical Level: The next higher level of abstraction describes the clinical concepts that are to be stored in the system. They can be represented in the form of archetypes and templates. The implementation of clinical concepts may involve physical-level structures. Users of logical level do not need to be aware of this complexity. Clinical domain experts use logical level.

iii) View Level: The highest level of abstraction describes only part of the entire EHR architecture. This corresponds to the service model. Several views are defined and the users see these views. In addition to hiding details of logical level, the views also provide a security mechanism to prevent users from accessing certain parts within EHR contents.

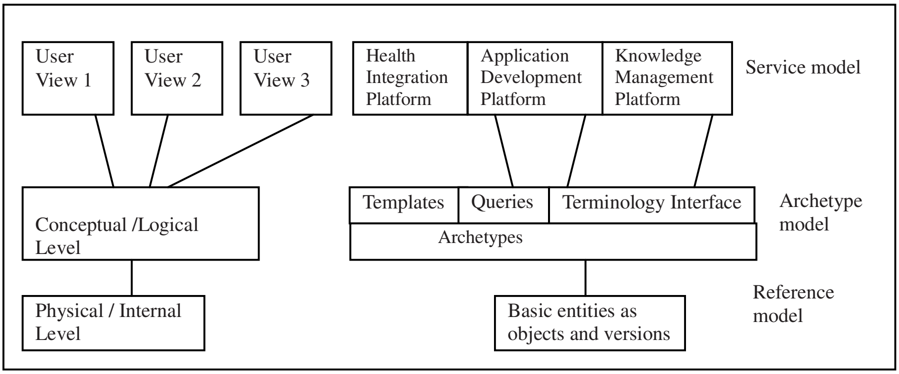

Fig. 4. DBMS architecture compared to openEHR Architecture

4 Semantic Interoperability in EHR: An Overview

In this section, we describe different levels of interoperability, the relationship of archetypes to semantic interoperability and Archetype Description language as a type of language for system level interactions (Figure 1). At the system level, the AQL language is supported for development of initial support infrastructure. In the following sections, this report presents how the application level can benefit from XML conversions and support of query language at application level, in the form of XQuery. Higher-level of support is an active area of research. Many research efforts aim to improve user interaction facilities [18] [22].

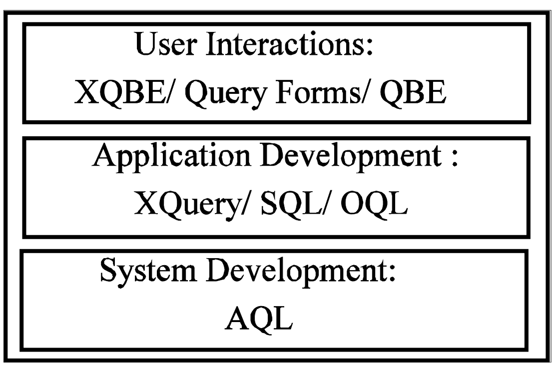

Fig. 5. Query Support at different Levels

4.1 Levels of Interoperability

There are three levels of interoperability [11]:

- Syntactic (data) interoperability
- Structural interoperability/Semantic interpretability.
- Semantic interoperability.

These are described in table 1.

Table 1. Levels of interoperability [11]

Levels of interoperability	Main mechanisms for interoperability	Description
Syntactic interoperability	openEHR Reference Model (RM)	The openEHR reference model alone ensures syntactic interoperability independent of any defined archetypes. The openEHR reference model does not define clinical knowledge. It is defined and communicated by archetypes, separately from the reference model. Hence, data items are communicated between systems only in terms of clearly defined, generic reference model instances. As the reference model is stable, achieving syntactic interoperability between systems is a simple task.
Structural interoperability	Archetypes	Structural interoperability is achieved by the definition and use of archetypes. As agreed models of clinical or other domain specific concepts, archetypes are clinically meaningful entities. An EHR entry (or a part) which has been archetyped will have the same meaning no matter where it appears. Thus, archetypes can be shared by multiple health systems and authorities, enabling information to be shared between different systems and types of healthcare professionals. Clinical knowledge can be shared and clinical information can be safely interpreted by exchanging archetypes.
Semantic interoperability	Domain Knowledge Governance	The use of archetypes and the reference model alone do not guarantee that different EHR systems and vendors will construct equivalent EHR extracts, and use the record hierarchy and terminology in consistent ways. For semantically interoperable systems, archetype development must be coordinated through systematic “Domain Knowledge Governance” tool. For example, it succeeds to avoid incompatible, overlapping archetypes for essentially the same concept.

4.2 Archetypes and Semantic Interoperability

Archetypes specify the design of the clinical data that a Health Care Professional needs to store in a computer system. Archetypes enable the formal definition of clinical content by domain experts without the need for technical understanding. These conserve the meaning of data by maintaining explicitly specified and well-structured clinical content for semantic interpretability. These can safely evolve and thus deal with ever-changing health knowledge using a two-level approach.

In simpler terms, an archetype is an agreed formal and interoperable specification of a re-usable clinical data set which underpins an electronic health record (EHR). It captures as much information about a particular and discrete clinical concept as possible. An example of a simple archetype is WEIGHT, which can be used in multiple places, wherever is required within an EHR. Once the format of an archetype is agreed and published, then it will be held in a 'library'(such as, clinical knowledge manager) and made available for use in any part of a given application, by multiple vendor systems, multiple institutions, and multiple geographical regions. Each group or entity using the same archetype will be able to understand and compute data captured by the same archetype in another clinical environment.

Archetypes are the basis for knowledge based systems (KBS) as these are means to define clinical knowledge (Figure 2). These are language neutral. These should be governed with an international scope, and should be developed by clinical experts in interdisciplinary cooperative way [13]. The developed archetypes need to be reviewed by other clinical experts (e.g., clinical review board) to ensure their completeness and relevance to evidence-based clinical practice. The archetype repository is a place of development, governance and primary source of archetypes. High quality archetypes with high quality clinical content are the key to semantic interoperability of clinical systems [13]. According to archetype editorial group (and clinical knowledge manager (CKM) [9]), the information should be sharable and computable.

Terminologies also help in achieving semantic interoperability. Terms within archetypes are linked (bound) to external terminologies like SNOMED-CT [11]. With the use of reference model, archetypes and a companion domain knowledge governance tool, the semantic interoperability of EHR systems becomes a reachable goal.

The openEHR foundation is developing archetypes which will ensure semantic interoperability. The openEHR archetypes are developed systematically through domain knowledge governance tools. According to the statistics provided by CKM, which is a domain knowledge governance tool, there are 227 numbers of archetypes [9]. Domain knowledge governance will ensure that archetypes will meet the information needs of the various areas.

With CKM, the users interested in modeling clinical content can participate in the creation and/or enhancement of an international set of archetypes. These provide the foundation for interoperable Electronic Health Records. CKM is a framework for managing archetypes. It helps in identifying which archetypes need to be standardized, and which are domain specific. It establishes a frame of reference and helps to train multidisciplinary teams for archetype development. The coordination effort team will inform and support domain knowledge governance. To support this, openEHR has employed the Web Ontology Language (OWL) and the Protégé OWL Plug-In to develop and maintain an Archetype Ontology which provides the necessary meta-information on archetypes for Domain Knowledge Governance. The Archetype Ontology captures the meta-information about archetypes needed to support Domain Knowledge Governance [11].

4.3 Archetype Definition Language (ADL)

Archetypes for any domain are described using a formal language known as Archetype deption language (ADL) [14]. ADL is path addressable like XML. The openEHR Archetype Object Model (AOM) describes the definitive semantic model of archetypes, in the form of an object model [15]. The AOM defines relationships which must hold true between the parts of an archetype for it to be valid as a whole. In simpler terms, all archetypes should conform to AOM. Since EHR has a hierarchical structure, ADL syntax is one possible serialisation of an archetype. ADL uses three other syntaxes, cADL (constraint form of ADL), dADL (data definition form of ADL), and a version of first-order predicate logic (FOPL), to describe constraints on data which are instances of RM [14].

The ADL archetype structure consists of archetype definition (expressed using cADL syntax), language, description, ontology, and revision_history (expressed using dADL syntax), invariant section (expressed using FOPL). The invariant section introduces assertions which relate to the entire archetype. These are used to make statements which are not possible within the block structure of the definition section. Similarly, the dADL syntax provides a formal means of expressing instance data based on an underlying information Model [14]. The cADL is a syntax which enables constraints on data defined by object-oriented information models to be expressed in archetypes or other knowledge definition formalisms [14].

Every ADL archetype is written with respect to a reference model. Archetypes are applied to data via the use of templates, which are defined at a local level. Templates [10] generally correspond closely to screen forms, and may be re-usable at a local or regional level. Templates do not introduce any new semantics to archetypes, they simply specify the use of particular archetypes, further compatible constraints, and default data values.

There are many parameters, such as weight, body temperature and heart rate in an EHR. The ADL for a parameter ‘Blood Pressure’ (BP) is shown in appendix A (also see Figure 2). ADL for other parameters are available at common repository [16]. ADL has a number of advantages over XML. It is both machine and human processable, and approximately, takes half space of XML. The leaf data type is more comprehensive set (including interval of numerics and date/time types). ADL adheres to object-oriented semantics that do not confuse between notions of attributes and elements. In ADL, there are two types of identifiers (from reference model) - the type names and attributes. Formally, it is possible to convert ADL into XML format and other formats [14]. Table 2 gives the comparison of ADL and XML.

In the near future, there is an important research issue regarding EHR systems, that is, whether all the archetype-based EHR systems will be created as ADL based database systems or ADL-enabled database systems (i.e., traditional database systems with enhanced ADL storage capabilities).

Table 2. Comparison between ADL and XML

Properties	ADL	XML
Machine Processable	Yes	Yes
Human Readable	Yes	Sometimes unreadable (e.g., XML-schema instance, OWL-RDF ontologies)
Leaf data Types	More comprehensive set, including interval of numerics and date/time types	String data; with XML Schema option- more comprehensive set
Adhering to object-oriented semantics	Yes, particularly for container types	XML schema languages do not follow object-oriented semantics
Representation of object properties	Uses attributes	Uses attributes and Sub-elements
Space (for storage)	Uses nearly half of space in XML	May have data redundancy

4.4 Interoperability and Different Levels of Interfaces

In Figure 1, the systems will use a XML/ADL type of language for system-to-system interactions. The healthcare worker will need an additional support layer. The existing support can be compared to (Figure 5)–

A) System Programmers level for development of EHR system- using ADL.

B) Application Programmer level for development of system applications, using XQuery, OQL (object query language) and SQL - type of interfaces (assuming the RDBMSs may support ADL in future).

C) Healthcare worker level interfaces: This is an active research area and no easy-to-use interfaces exist till date. In section 5.2, an attempt to provide one such interface has been outlined. It aims to demonstrate – how interoperability at application programmer level, can be made to support a user interface at healthcare worker level.

5 Querying Archetype Based EHR

EHRs allow multiple representations [17]. In principle, EHRs can be represented as relational structures (governed by an object/relational mapping layer), and in various XML storage representations. There are many properties and classes in the reference model, but the archetypes will constrain only those parts of a model which are meaningful to constrain. These constraints cannot be stronger than those in reference model. For example, if an attribute is mandatory in RM, it is not valid to express a constraint allowing the attribute to be optional in the archetype (ADL). So, the single ADL file is not sufficient enough for querying. The user may want to query some properties or attributes from RM, along with the querying from properties in archetypes. In order to create a data instance of a parameter of EHR, we need different archetypes in ADL, and also these archetypes may belong to different categories of archetypes.

For example, to create a data instance for Blood Pressure, we need two different archetypes-namely, encounter and blood pressure. These archetypes belong to different categories viz., COMPOSITION and OBSERVATION.

The different categories have different structure. At the time of query, a user faces this problem- which archetypes must be included in querying? For example, querying on BP requires the use of two archetypes viz., Encounter archetype (belonging to COMPOSITION category of RM) and Blood Pressure archetype (belonging to OBSERVATION category of RM). This problem can be addressed by the use of templates. Archetypes are encapsulated by Templates for the purpose of intelligent querying [10]. The templates are used for archetype composition or chaining. Archetypes provide the pattern for data rather than an exact template. The result of the use of archetypes to create data in the EHR is that the structure of data in any top-level object conforms to the constraints defined in a composition of archetypes chosen by a template.

At the user level, querying data regarding BP must be very simple. The user only knows BP as a parameter and will query that parameter only.

The EHR system must have an appealing and responsive query interface that provides a rich overview of data and an effective query mechanism for patient data. The overall solution should be designed with an end-to-end perspective in mind. A query interface is required that will support users at varying levels of query skills. These include semi-skilled users at clinics or hospitals.

5.1 Archetype Query Language (AQL)

To query upon EHRs, a query language, Archetype Query Language (AQL) has been developed [8]. It is neutral to EHRs, programming languages and system environments. It depends on the openEHR archetype model, semantics and its syntax. AQL is able to express queries from an archetype-based EHR system. The use of AQL is confined to a skilled programmers' level. It was first named as EQL (EHR Query Language) which has been enhanced with the following two innovations [17]:

i) utilizing the openEHR path mechanism to represent the query criteria and the response or results; and

ii) using a 'containment' mechanism to indicate the data hierarchy and to constrain the source data to which the query is applied.

The syntax of AQL is illustrated by the help of example.

Query: Find all blood pressure values, where systolic value is greater than (or equal to) 140, or diastolic value is greater than (or equals to) 90, within a specified EHR.

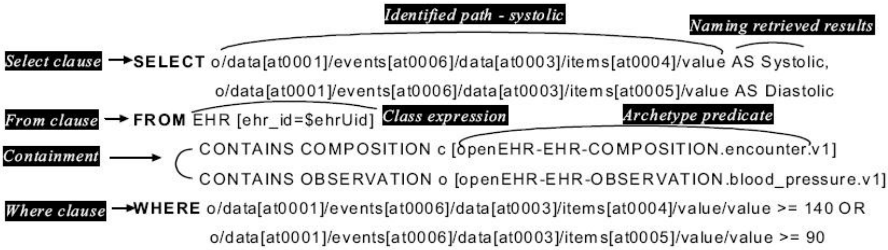

Fig. 6. Syntax of AQL

5.2 High-Level Database Query Interfaces

AQL is difficult for semi-skilled users. It requires the knowledge of archetypes and knowledge of languages such as ADL, XML and SQL. At the present moment, there is no easy-to-use query language interface available for EHRs database. We propose to study for the convenience of healthcare professionals a high-level interface for querying archetype based EHR systems based on the proposed query interface XQBE [18]. An alternative approach proposed by Ocean informatics [19] suggests using a query builder tool, to construct AQL query. It requires form related inputs and more skills on the part of the user. Our goal is similar and it is easy to achieve with the help of XQBE.

XQBE [18] is a user-friendly, visual query language for expressing a large subset of XQuery in a visual form. Its simplicity, expressive power and direct mapping to XQuery are some of the highlighting features for its use. Like XQuery, XQBE relies on the underlying expressions in XML. It requires all data to be converted to XML form. It presents a user with XML sub-tree expressions for the items of user interests. XQBE's main graphical elements are trees. There are two parts, the source part which describes the XML data to be matched against the set of input documents, and the construct part, which specifies which parts will be retained in the result, together with (optional) newly generated XML items.

In order to adopt a XQBE like interface at user level, we propose to convert ADL into XML. ADL can be mapped to an equivalent XML instance. ADL is hierarchical in nature and has a unique identification to each node. Thus, paths are directly convertible to XPath expressions. These can be created. According to Beale and Heard [6], the particular mapping chosen may be designed to be a faithful reflection of the semantics of object-oriented data. There may be need for some additional tags for making the mapping of nested container attributes since XML does not have a systematic object-oriented semantics. Thus, single attribute nodes can be mapped to tagged nodes of the same name. Container attribute nodes map to a series of tagged nodes of the same name, each with the XML attribute 'id' set to the node identifier. Type names map to XML 'type' attributes.

In the present proposal, the patient data description is converted to XML form [21]. It is suitably reformed for adoption of XQBE interface. Thus users can directly use XQBE query interface to access patient data. This process eliminates the need to learn and use the AQL language on the part of the users [21]. The XQBE skills can be learnt with ease [18].

5.3 Mapping ADL to XQBE for EHR Data

Database queries are usually dependent on local database schemas but archetype systems being proposed aim to have portable queries. The queries play a crucial role in decision support and epidemiological situations. The XQBE approach for archetype-based EHRs is being proposed for semi-skilled users (such as doctors, physicians, nurses). The mapping process to create XQBE is shown in following steps (Figure 7).

i) The conversion of ADL file into XML file.
ii) Generation of DTD for the XML file.
iii) Generation of XQBE interface structure.

Subsequently, for the semi-skilled user, this three step process will facilitate in querying archetype based EHRs. The step (ii) in above process will aid in the guided construction of query provided by XQBE [18]. However, for some users (skilled in use of XML) the step (ii) may not be needed and XQBE can be used directly for the XML file.

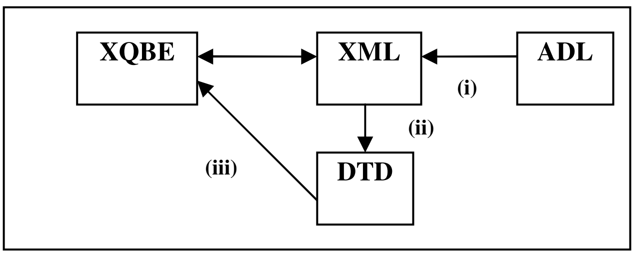

Fig. 7. Mapping process to present XQBE for EHRs [21]

Query Scenario1. Find all Blood Pressure values, having the systolic_BP and diastolic_BP, (where systolic_BP >= 140 and diastolic_BP >=90).

The AQL syntax for the above query is shown in appendix B. By using XQBE approach for querying, we perform step (i) to step (iii) as explained, on BP parameter. For each case of query, and for querying different parameters of EHR, we need to convert each parameter (in form of adl) to a corresponding xml for the demonstration. We propose to develop an automated tool in the subsequent phase. The clinical user will be provided with a substituted XQBE interface (Figure 9) in place of AQL.

XQBE is a visual interface. A user is presented with graphical image of EHR components, for example, blood Pressure (BP) in this case. In Figure 9, based on the selected source data, the user defines a target sub-tree (in XML form) to express the query (and its outcome). The query is expressed by using the graphical elements of XQBE [18]. The source part of the query is built using the DTD. A guided construction is provided to the user to add predicates for the query. The construct (or result) part of the query is built by the user using the graphical elements of XQBE by dragging and dropping them.

```
<!ELEMENT adl_version ( #PCDATA ) >
<!ELEMENT archetype ( original_language, description,
archetype_id, adl_version, concept, definition, ontology ) >
<!ELEMENT archetype_id ( value ) >
<! ELEMENT assumed_value (terminology_id,
code_string, magnitude?, units?, precision? ) >
<! ELEMENT attributes (rm_attribute_name, existence,
children+, cardinality? ) >
<! ATTLIST attributes xsi: type
(C_MULTIPLE_ATTRIBUTE | C_SINGLE_ATTRIBUTE)
#REQUIRED >
<! ELEMENT cardinality (is_ordered, is_unique, interval)
>
<! ELEMENT children (assumed_value | attributes |
code_list | includes | item | list | node_id | occurrences |
property | rm_type_name | target_path | terminology_id)* >
<! ATTLIST children xsi: type
(ARCHETYPE_INTERNAL_REF | ARCHETYPE_SLOT |
C_CODE_PHRASE | C_COMPLEX_OBJECT |
C_DV_QUANTITY | C_PRIMITIVE_OBJECT)
```

Fig. 8. A sample of BP.dtd

Figure 9 shows the element nodes and subelement nodes in the source part. The subelements (systolic and diastolic) of the BP element, one systolic and one diastolic satisfy condition1 (systolic>=140) AND condition2 (diastolic>=90) are described with the help of XQBE convention. As per the convention, an arc containing '+' indicates that 'children' element node may exist at any level of nesting (as in Xpath we use '//'). The construct part consists of element node for BP (set tag T-node), and also element nodes for systolic and diastolic, which relates the projected BP element nodes to its systolic and diastolic subelements. The fragment node (shown by filled triangle) indicates that the entire fragment of systolic and diastolic must be retained in the result.

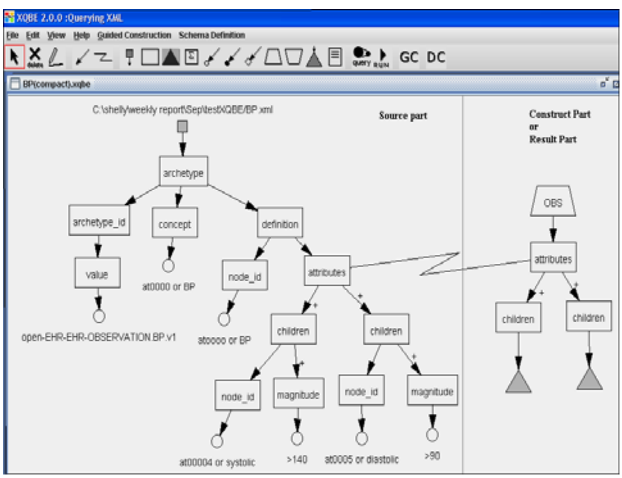

Fig. 9. BP.XQBE- an XQBE template for query

6 Discussions

There are several concerns regarding semantic interoperability which are listed below:

- How will the legacy migration of electronic health record systems to dual model approach based EHR systems be achieved?
- With evolution of domain clinical knowledge concepts, is it possible to achieve semantic interoperability in next 5 years or 10 years?
- At present, few people are trained to work at the intersection of biomedicine and IT. Will clinical domain experts really be interested in the development of archetypes, which are keystones in achieving semantic interoperability?
- Archetype Query Language [8] has been developed by openEHR for querying EHR data. It helps in semantic interoperability. Will it really be a powerful query language?
- How much cost will be involved for maintaining EHR systems?
- In near future, will archetype based EHR systems be semantically interoperable?

Sharing and exchange of knowledge, resources and information pertaining to the care of patients is needed by healthcare professionals. Archetypes and templates are a paradigm for building semantically-enabled software systems. Thus, archetype based EHR system ensures semantic interoperability. Database queries, usually dependent on local database schemas but archetype systems being proposed aim to have portable queries.

7 Summary and Conclusions

Existing query language interfaces are not suitable for healthcare applications. A proposal for developing high level interface suitable for healthcare users has been presented. Further, high-level user facilities, such as QBE sort of interface for ADL using the graphical elements of XQBE [18],or other existing high-level interfaces, such as XGI (A graphical interface for XQuery creation) [23], and GLASS (A Graphical Query Language for semi-structured data) [24] are aimed to be analyzed. Due to the time constraints, the research efforts for these proposals have been omitted. Finally, we conclude the paper with pointing out the strong need of high-level query interfaces for healthcare professionals because of heterogeneous nature and fragmentation of healthcare organizations. Our aim is to support a healthcare user at varying levels to enhance semantic interoperability.

References

1. ISO/TC 215 Technical Report: Electronic Health Record Definition, Scope, and Context. Second Draft (August 2003)
2. Institute of Electrical and Electronics Engineers (IEEE). IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries, New York, NY (1990)

3. European committee for Standardization, Technical committee on Health informatics, Standard for EHR communication, http://www.cen.eu
4. http://www.openehr.org
5. ISO 13606-2, Health informatics - Electronic health record communication - Part 2: Archetype interchange specification (2008)
6. Beale, T., Heard, S.: openEHR Architecture: Architecture Overview in The openEHR release 1.0.2. In: Beale, T., Heard, S. (eds.) openEHR Foundation (2008)
7. Silberschatz, K., Sudarshan, S.: Database Systems Concepts, 4th edn. McGraw-Hill, New York (2002) international edition
8. ArchetypeQueryLanguage, http://www.openehr.org/wiki/display/spec/Archetype+Query+Language+AQL+Ocean
9. Clinical Knowledge Manager version 1.0.5, http://www.openehr.org/knowledge
10. Beale, T., Heard, S.: Archetype Definitions and Principles in The openEHR foundation release 1.0.2. In: Beale, T., Heard, S. (eds.) The openEHR foundation (2005)
11. Garde, S., Knaup, P., Hovenga, E.J.S., Heard, S.: Towards Semantic Interoperability for Electronic Health Records: Domain Knowledge Governance for openEHR Archetypes. Methods of Information in Medicine (2007)
12. ISO 13606-1, Health informatics - Electronic health record communication - Part 1: Reference Model (2008)
13. Garde, S., Hovenga, E.J.S., Gränz, J., Foozonkhah, S., Heard, S.: Managing Archetypes for Sustainable and Semantically Interoperable Electronic Health Records. Electronic Journal of Health informatics (2007)
14. Beale, T., Heard, S.: The openEHR Archetype Model-Archetype Definition Language ADL 1.4. openEHR release 1.0.2, Issue date December 12 (2008)
15. Beale, T.: The openEHR Archetype Model-Archetype Object Model. openEHR release 1.0.2, Issue date November 20 (2008)
16. http://www.openehr.org/svn/knowledge/archetypes/dev/html/index_en.html (ADL for archetypes) (accessed 04/12/2009)
17. Chunlan, M., Heath, F., Thomas, B., Sam, H.: EHR Query Language (EQL)-A Query Language for Archetype-Based Health Records. In: MEDINFO 2007 (2007)
18. Braga, D., Campi, A., Ceri, S.: XQBE (XQueryBy Example): A Visual Interface to the Standard XML Query Language. ACM Transactions on Database Systems 30(2), 398–443 (2005)
19. http://www.oceaninformatics.com/Solutions/ocean-products/Clinical-Modelling/Ocean-Query-Builder.html (Query builder, Ocean Informatics) (accessed 10/12/2009)
20. http://dbgroup.elet.polimi.it/xquery/XQBEDownload.html (XQBE 2.0.0)
21. Sachdeva, S., Bhalla, S.: Implementing High-Level Query Language Interfaces for Archetype-Based Electronic Health Records Database. In: International Conference on Management of Data (COMAD), December 9-12 (2009)
22. Jayapandian, M., Jagadish, H.V.: Automating the Design and Construction of Query Forms. IEEE Transactions on Knowledge and Data Engineering 21(10), 1389–1402 (2009)
23. Li, X., Gennari, J.H., Brinkley, J.F.: XGI: A Graphical Interface for XQuery Creation. In: AMIA 2007 Symposium Proceedings, pp. 453–457 (2007)
24. Ni, W., Ling, T.W.: GLASS: A Graphical Query Language for semi-structured data. In: Proceedings of the Eighth International Conference on Database Systems for Advanced Applications (DASFAA 2003). IEEE, Los Alamitos (2003)
25. WHO. WHA58.28 - Resolution on eHealth. 58th World Health Assembly (2005), http://www.who.int/gb/ebwha/pdf_files/WHA58/WHA58_28-en.pdf

Appendix A

A brief sample of ADL for Blood Pressure Archetype
(BP.adl downloaded from [9])

```
definition
OBSERVATION [at0000] matches {        -- Blood pressure
data matches {
  HISTORY[at0001] matches {  -- history
     events cardinality matches {1..*; unordered} matches {
        EVENT[at0006] occurrences matches {0..*} matches {   -- any event
           data matches {
              ITEM_TREE[at0003] matches {          -- blood pressure
              items cardinality matches {0..*; unordered} matches {
  ELEMENT[at0004] occurrences matches {0..1} matches      -- Systolic
               value matches {
                C_DV_QUANTITY <
                        property = <[openehr::125]>
                    list = <  ["1"] = <
                       units = <"mm[Hg]">
                    magnitude = <|0.0..<1000.0|>
                    precision = <|0|>
                               >  >  >
                             }}
  ELEMENT[at0005] occurrences matches {0..1} matches {             -- Diastolic
               value matches {
                   C_DV_QUANTITY <
                     property = <[openehr::125]>
                     list = <            ["1"] = <
                         units = <"mm[Hg]">
                      magnitude = <|0.0..<1000.0|>
                       precision = <|0|>
                         >  >  >        } }
 ELEMENT[at1006] occurrences matches {0..1} matches {-- Mean Arterial Pres-
sure
          value matches {
                C_DV_QUANTITY <
                        property = <[openehr::125]>
                        list = < ["1"] = <
                        units = <"mm[Hg]">
                        magnitude = <|0.0..<1000.0|>
                        precision = <|0|>
                        >  >  >           } }
 ELEMENT[at1007] occurrences matches {0..1} matches { -- Pulse Pressure
          value matches {
                C_DV_QUANTITY <
                        property = <[openehr::125]>
```

```
                    list = < ["1"] = <
               units = <"mm[Hg]">
               magnitude = <|0.0..<1000.0|>
             precision = <|0|>
               > > >        } }
ELEMENT[at0033] occurrences matches {0..1} matches {   -- Comment
      value matches {
        DV_TEXT matches {*}
                   }}}}}
state matches {
      ITEM_TREE[at0007] matches {   -- state structure
       items cardinality matches {0..*; unordered} matches {
 ELEMENT[at0008] occurrences matches {0..1} matches {   --  Position
         value matches {
             DV_CODED_TEXT matches {
            defining_code matches {
               [local::
                    at1000,          -- Standing
             at1001,          -- Sitting
             at1002,          -- Reclining
             at1003,          -- Lying
             at1013,          -- Trendelenburg
             at1014;          -- Left Lateral
             at1001]          -- assumed value
                      } } } }
```

Appendix B

AQL Syntax for BP Query scenario1

```
SELECT obs/data[at0001]/events[at0006]/data[at0003]/
items[at0004]/value/magnitude, obs/data[at0001]/events
[at0006]/data[at0003]/items[at0005]/value/magnitude
FROM EHR [ehr_id/value=$ehrUid] CONTAINS
COMPOSITION [openEHR-EHR-COMPOSITION .encounter.v1]
CONTAINS OBSERVATION obs openEHR-EHR-OBSER
VATION.blood_pressure.v1]
WHERE obs/data[at0001]/events[at0006]/
data[at0003]/items[at0004]/value/magnitude>= 140 AND
obs/data[at0001]/events[at0006]/data[at0003]/items[at0005]/value/
magnitude>=90
```